CDT-D^{2}AIR Course on
D^2 Robots and Autonomous Agents

Safe Development – Some Issues and Approaches

Subramanian Ramamoorthy
School of Informatics
University of Edinburgh

24 April 2025

Topics around a Theme

- Understanding accidents and notions of safety
- Safety cases and approaches to assurance
- Responsibility, “blame” and implications for safety monitoring

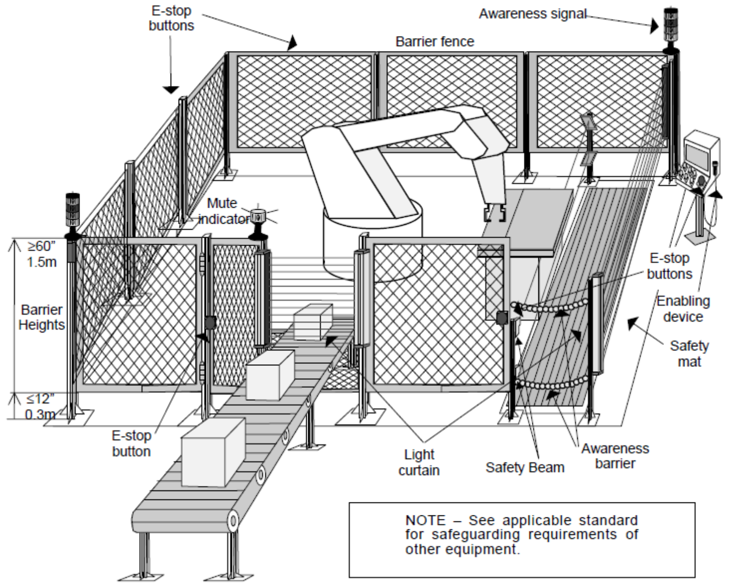
Understanding Accidents and Notions of Safety

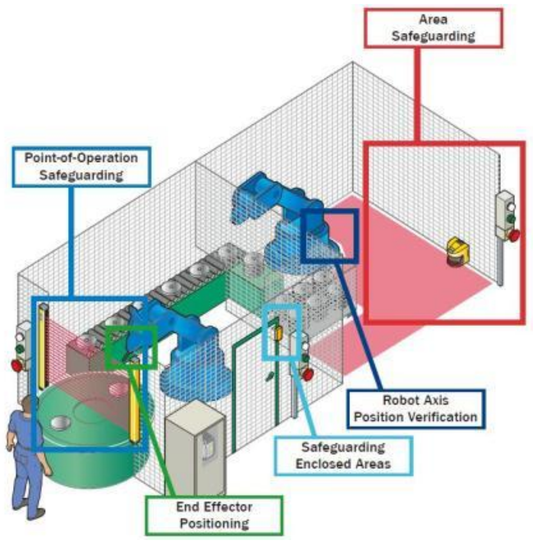
Why? Example 1

[Source: aslib.co.uk]

[Source: www.iff.fraunhofer.de]

Traditional Notions of Robot Safety





Area Safeguarding
Point-of-Operation Safeguarding
Robot Axis Position Verification
Safeguarding Enclosed Areas
End Effector Positioning

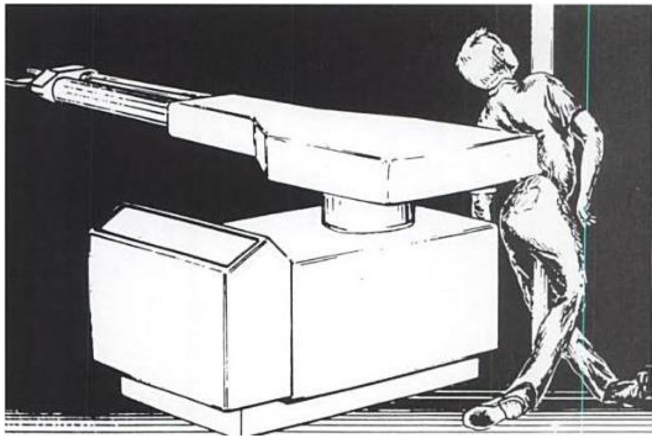
Deployed robot safety systems

Robots, depending on the task, may generate paint mist, welding fumes, plastic fumes, etc. In general, the robot, on occasion is used in environments or tasks too dangerous for workers, and as such creates hazards not specific to the robot but specific to the task.

[Source: G. Cui et al., Ontario]

Historical Examples of Accidents

Example 1: First fatal robot-related accident in the U.S.



On July 21, 1984, a die cast operator was working with an automated die cast system utilizing a Unimate Robot, which was programmed to extract the casting from the die-cast machine, dip it into a quench tank and insert it into an automatic trim press.

A neighboring employee discovered the victim pinned between the right rear of the robot and a safety pole in a slumped but upright position. The victim died five days later in the hospital.

[Source: G. Cui et al., Ontario]

Historical Examples of Accidents

Example 2:

A material handling robot was operating in its automatic mode and a worker violated safety devices to enter the robot work cell. The worker became trapped between the robot and a post anchored to the floor, was injured and died a few days later.

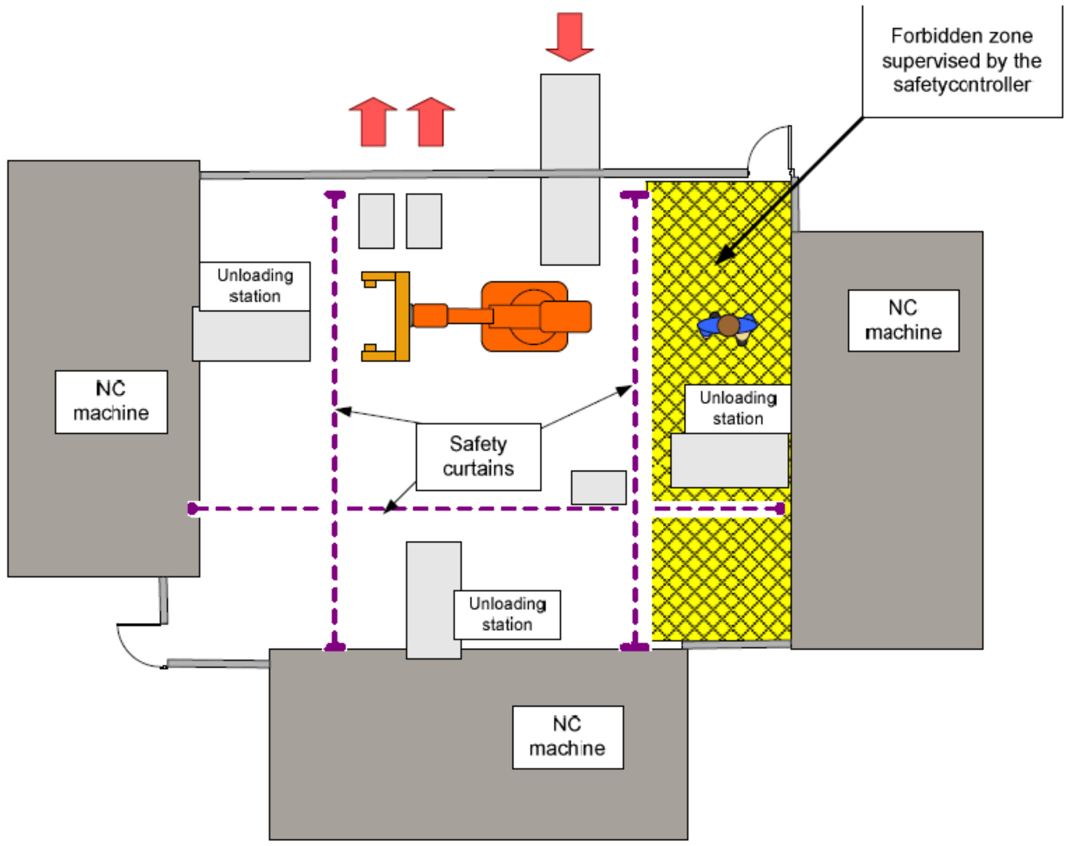
Example 3:

A maintenance person climbed over a safety fence without turning off power to a robot and performed tasks in the robot work zone while it was temporarily stopped. When the robot recommenced operation, it pushed the person into a grinding machine, killing the person.

[Source: G. Cui et al., Ontario]

How could ‘Safety’ be Implemented?

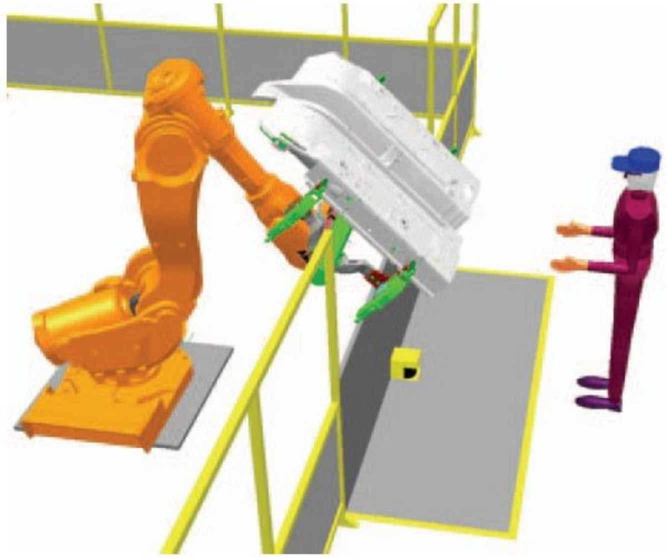
Example 1: Monitor and increase safety of tool zones



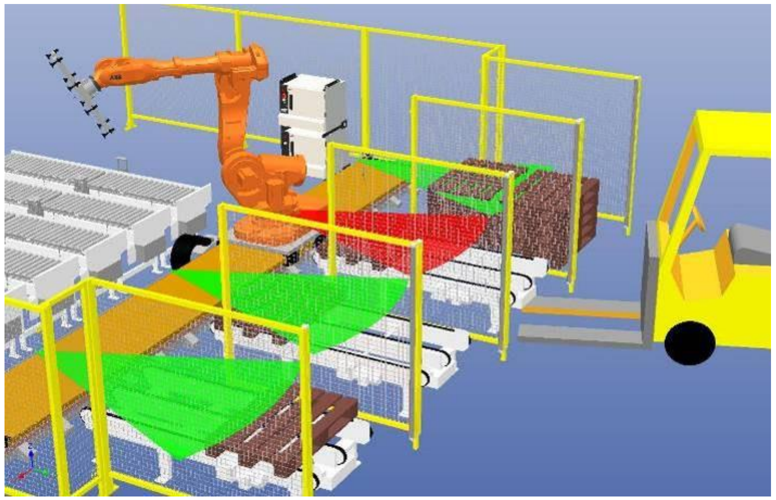
[Source: G. Cui et al., Ontario]

How could ‘Safety’ be Implemented?

Example 2: Safe stand still/direct loading of a robot



Example 3: Safe axis ranges with track motions



[Source: G. Cui et al., Ontario]

Characterizing an *Unsafe* Robot

Human Interaction

Control Errors

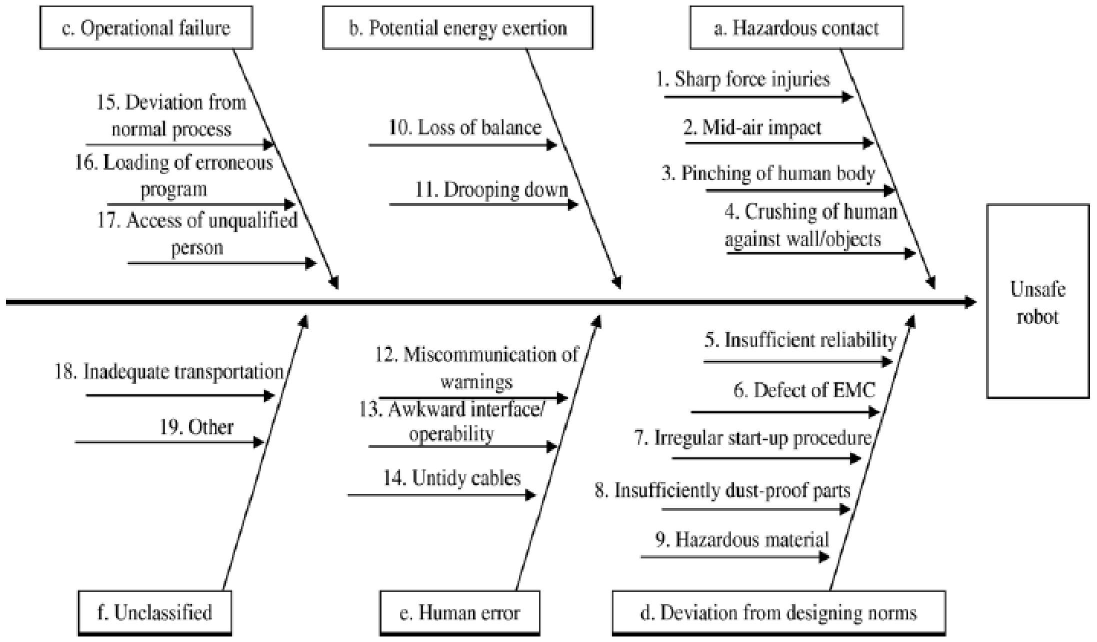
Unauthorized Access

Mechanical Failures

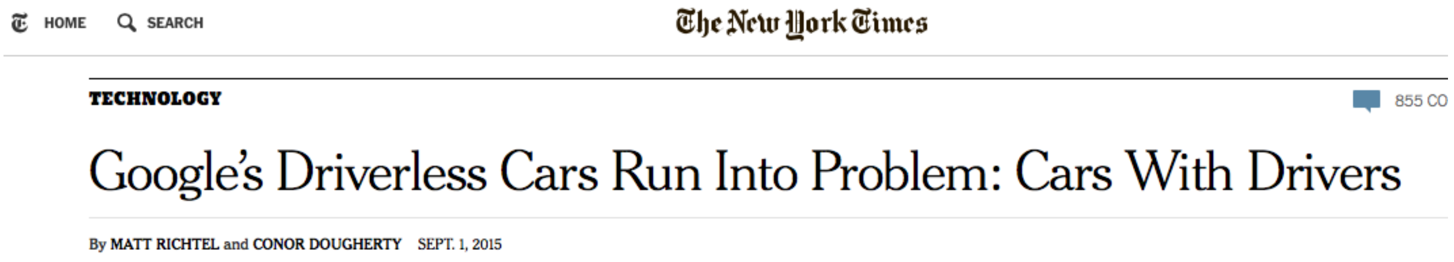
Environmental Sources

Power Systems

Improper Installation


c. Operational failure
b. Potential energy exertion
a. Hazardous contact
15. Deviation from normal process
16. Loading of erroneous program
17. Access of unqualified person
10. Loss of balance
11. Drooping down
1. Sharp force injuries
2. Mid-air impact
3. Pinching of human body
4. Crushing of human against wall/objects
Unsafe robot
18. Inadequate transportation
19. Other
12. Miscommunication of warnings
13. Awkward interface/ operability
14. Untidy cables
5. Insufficient reliability
6. Defect of EMC
7. Irregular start-up procedure
8. Insufficiently dust-proof parts
9. Hazardous material
f. Unclassified
e. Human error
d. Deviation from designing norms

Why? Example 2

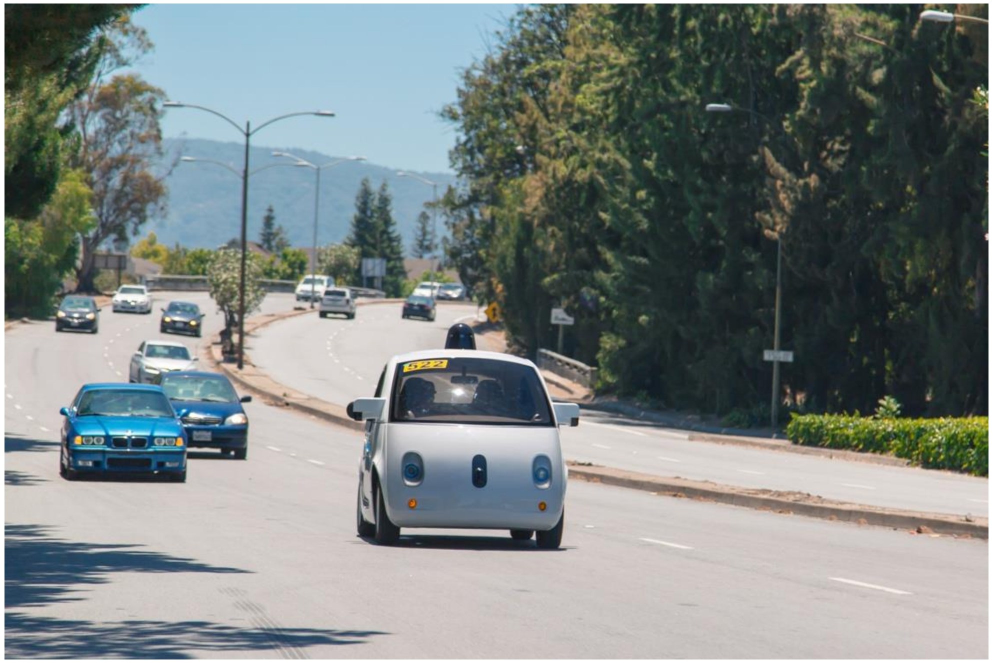


HOME SEARCH The New York Times

TECHNOLOGY 855 CO

Google's Driverless Cars Run Into Problem: Cars With Drivers

By MATT RICHTEL and CONOR DOUGHERTY SEPT. 1, 2015



Are these Issues Unique to Robotics?

NO!

- Many other engineering systems have been through a similar path towards understanding safety
- Avionics, maritime systems, nuclear reactors, …
- ... office printers!

- Many famous examples of failures which are systemic rather than individual component driven

Perrow's Notion of *Normal Accidents*

- While many initial accident analyses have blamed the human operators, the real fault lies in *system* design
- Certain high-risk systems, because of the way they configure sequences of subsystems, are *naturally* prone to eventually resulting in an accident.
- So, Three Mile Island was a Normal Accident

(what is robotics equivalent?)

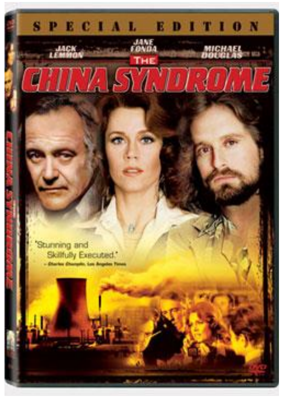
Normal Accidents: Core Argument

- Interactive Complexity
  - Failures of two components interact in an unexpected way
- Tightly Coupled
  - Processes that are parts of a system that happen quickly and cannot be turned off or isolated

- Perrow's Thesis: Tightly coupled systems with high interactive complexity will **necessarily** have Normal Accidents

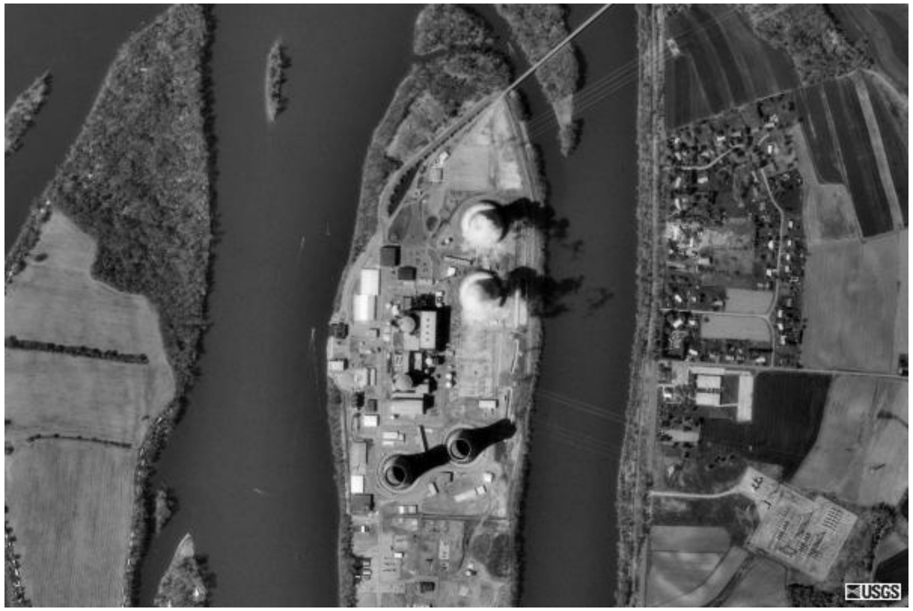
Case Study Example: Three Mile Island

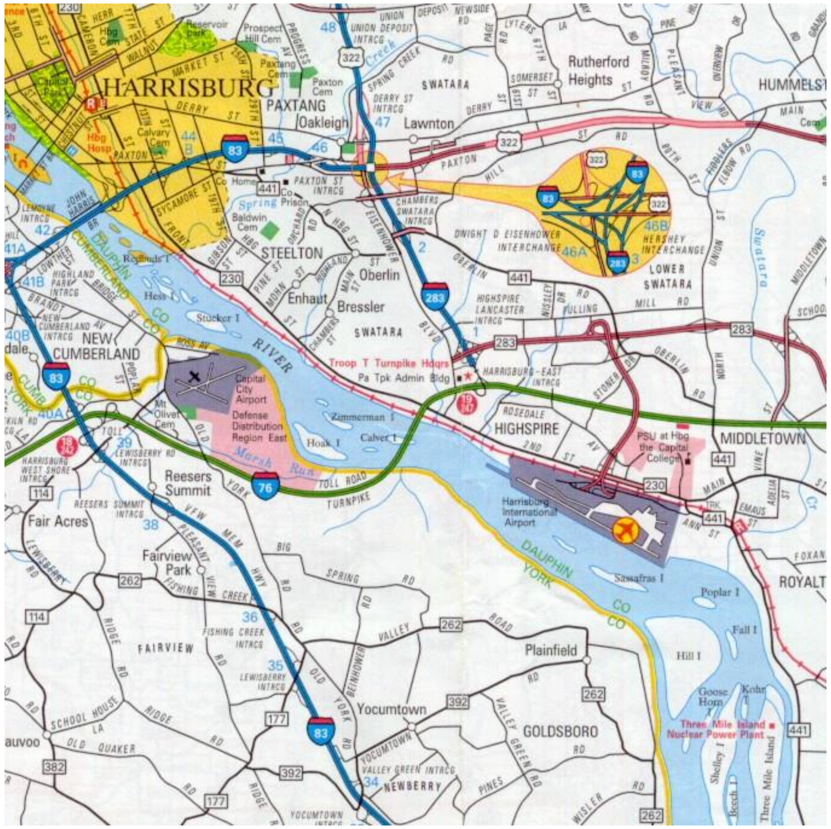
- Perhaps the most famous nuclear accident in the US
- On March 16, 1979, the movie China Syndrome (addresses social issues around nuclear accidents) is released
- 12 days later, March 28, 1979, the worst civilian nuclear accident in the US occurred at the Three Mile Island Nuclear Power Plant on the Susquehanna River, south of Harrisburg, PA.



[Source: Michael Carini, astro.wku.edu]

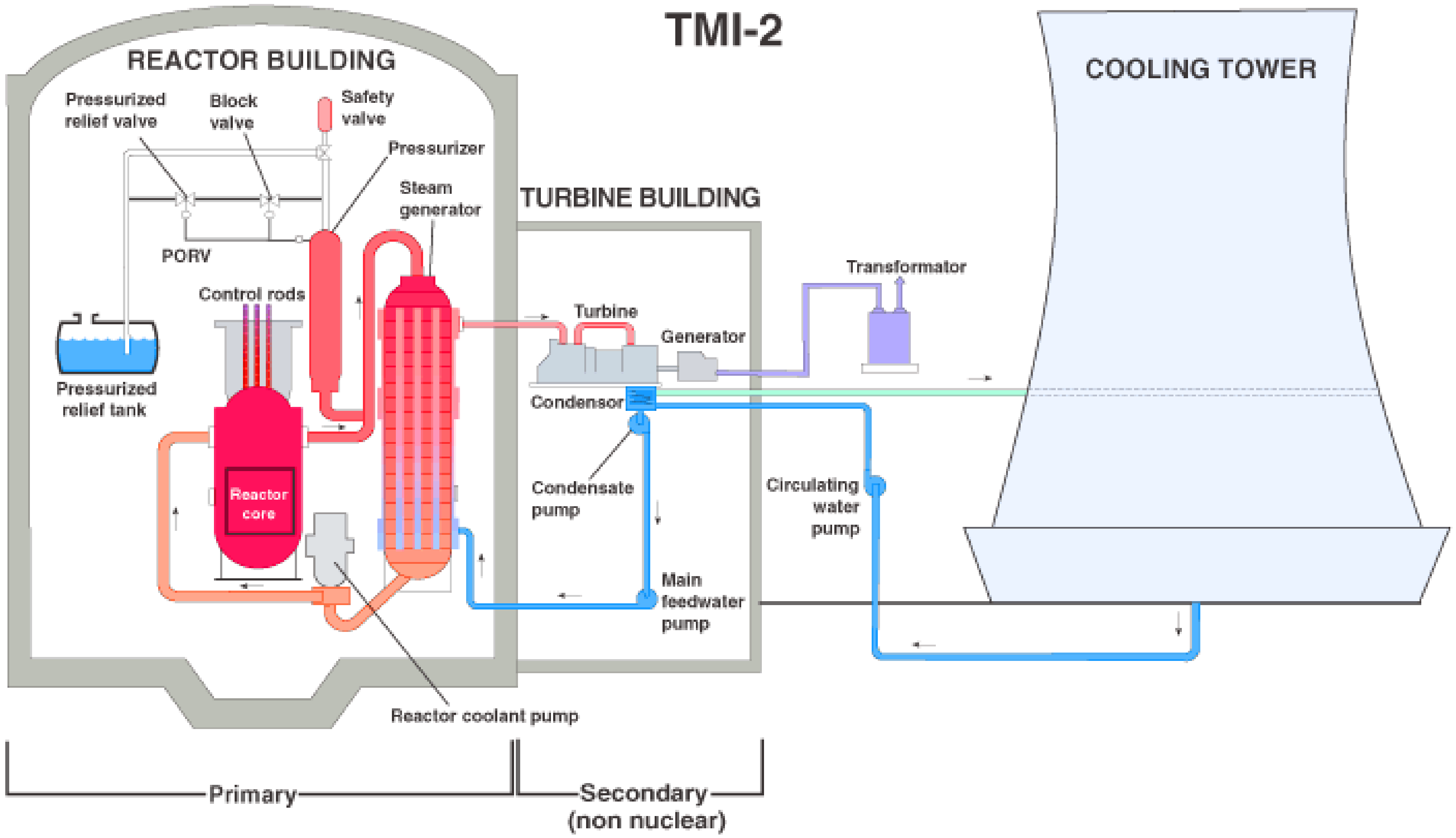
Three Mile Island: Location





[Source: Michael Carini, astro.wku.edu]

Example: Three Mile Island



Cooling System Setup

- Primary Cooling System
  - High pressure, radioactive, water circulating through the reactor.
  - Heat Exchanger transfers heat to the secondary system

- Secondary Cooling System
  - Cools the primary cooling system
  - Creates steam to run the turbines to generate electricity
  - Due to thin tubes in the turbine it must be very pure Continuously cleaned by a "polisher system"

A Sequence of Events

- The polisher leaked about a cup a day of water through a seal
- Water vapor got into a pneumatic system that drives some instruments
- This water vapor interrupted pressure to two valves in the feedwater system, which caused two feedwater pumps to shut down
- Lack of flow in the secondary system triggered a safety system that shut down the turbines
- This was the first indication of trouble to the operators
- At this point the reactor still needs to be cooled – or else

Sequence of Events: Emergency System

- An emergency feedwater system starts up to pump stored cold water through the secondary system to remove the accumulating heat
- The pumps were running, but valves on the pipes were incorrectly left closed from prior maintenance
- The operators insist they were left open; checklist says so
- A Repair Tag on a broken indicator hung over the indicator on the control panel that indicated that the valves were closed
- Redundant pipes, redundant pumps, and redundant valves, all thwarted by having the two valves physically at the same place and mis-set
- Eight minutes later they noticed they were shut by then the damage was done

No Cooling = Reactor Heats Up

- Due to overheating the reactor "scrammed” automatically
- This shuts down the reaction
- Enough heat remains in the reactor to require at normal working several days to cool off
- Without cooling the pressure goes up
- An ASU Automatic Safety Device takes over to temporarily relieve the pressure: the Pilot Operated Relief Valve (PORV)

PORV (Pilot Operated Relief Valve)

- The PORV is supposed to vent pressure briefly, and then reclose
- If it stays open too long liquid escapes, pressure in the reactor drops, steam forms causing voids in the water, cooling is impaired and some places get yet hotter
- Thirty-two thousand gallons of water eventually went out this unclosed valve
- There was an indication on the control panel that the message to reseat had been sent to the valve
- However, no indication was available that it had reseated
- We are now thirteen seconds into the "transient"
- An indicator shows that there is extra water from an unknown source

Automatic Cooling Pump Starts

- This is another automatic safety system that pumps water to cool the reactor automatically starts at 13 seconds. The second was manually started by the operator
- For three minutes it looked like the core was being cooled successfully
- However, apparently due to the steam voids, the cooling was not happening
- The secondary steam generators were not getting water and boiled dry - at the same time water was flowing out of the primary cooling system through the stuck pressure relief valve

High Pressure Injection Starts

- This is an automatic emergency device that forces cold water into the reactor to cool it down.
- The reactor was flooded for two minutes, and then the operators drastically cut back the flow. This was regarded as the key operator error; what they did not realize was that the water was flowing out the PORV and the core would become uncovered
- Two dials confused the operators:
  - one said the pressure in the reactor was rising
  - the other said it was falling
- The Kemeny commission thought the operators should have realized this meant LOCA (Loss of Coolant Accident)

What is it Like in Control Room?

- Three audible alarms are making a din
- Many of the 1,600 indicator lights are blinking

- The computer is way behind in printing out error messages
- It turns out they can only be printed, not spooled to disk, to see the current condition they would have to purge the printer and loose potentially valuable information

- The reactor coolant pumps begin the bang and shake, due to cavitation from lack of water to pump-they are shut off

Stuck Open PORV Valve Discovered

- The operators checked the valve and found it open
- They closed it
- With some trepidation since they were messing with a safety system
- The reactor core had been uncovered at this point and had partially melted
- Another 30 minutes without coolant and it would probably have been a total melt down (e.g., Chernobyl)

Tell tale Signs

- The whole system is never all up and working as designed thus it is hard to understand
- When things start to fail the system is even harder to understand
- Safety systems are not always working
  - some are down, and known to be
  - some are accidentally turned off
  - some are not set properly
  - others fail to work when needed
- There are often not direct indicators of what is happening operators figure it out indirectly

Can this happen elsewhere? With/to robots?

Tell tale Signs

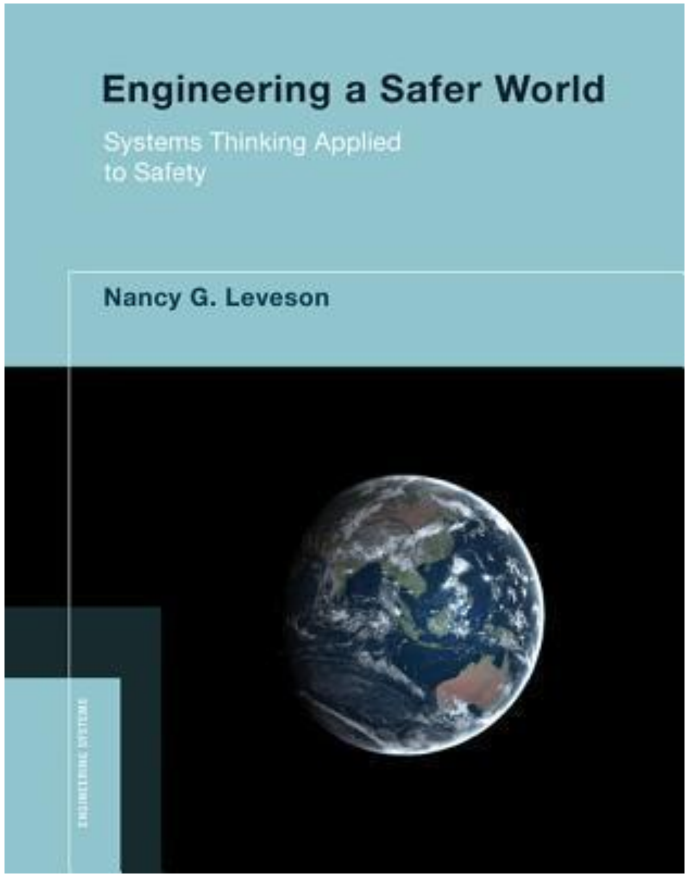
- The whole system is never all up and working as designed thus it is hard to understand
- When things start to fail the system is even harder to understand
- Safety systems are not always working
  - some are down, and known to be
  - some are accidentally turned off
  - some are not set properly
  - others fail to work when needed
- There are often not direct indicators of what is happening operators figure it out indirectly

Can this happen elsewhere? With/to robots?

Systems Thinking Applied to Safety

Questioning assumptions:

- High reliability is neither necessary nor sufficient for safety
- Accidents are complex processes involving entire sociotechnical systems. Traditional event-chain models cannot describe the process adequately.
- Risk and safety may be best understood and communicated in ways other than probabilistic risk analysis


Engineering a Safer World

Systems Thinking Applied to Safety

Nancy G. Leveson

ENGINEERING SYSTEMS

Systems Thinking Applied to Safety

- Operator behaviour is a product of the environment in which it occurs – must change environment to reduce “error”
- Highly reliable software is not necessarily safe
- Systems will migrate towards states of higher risk. Such migration is predictable and preventable by appropriate system design, or predictable using leading indicators.
- “Blame is the enemy of safety”

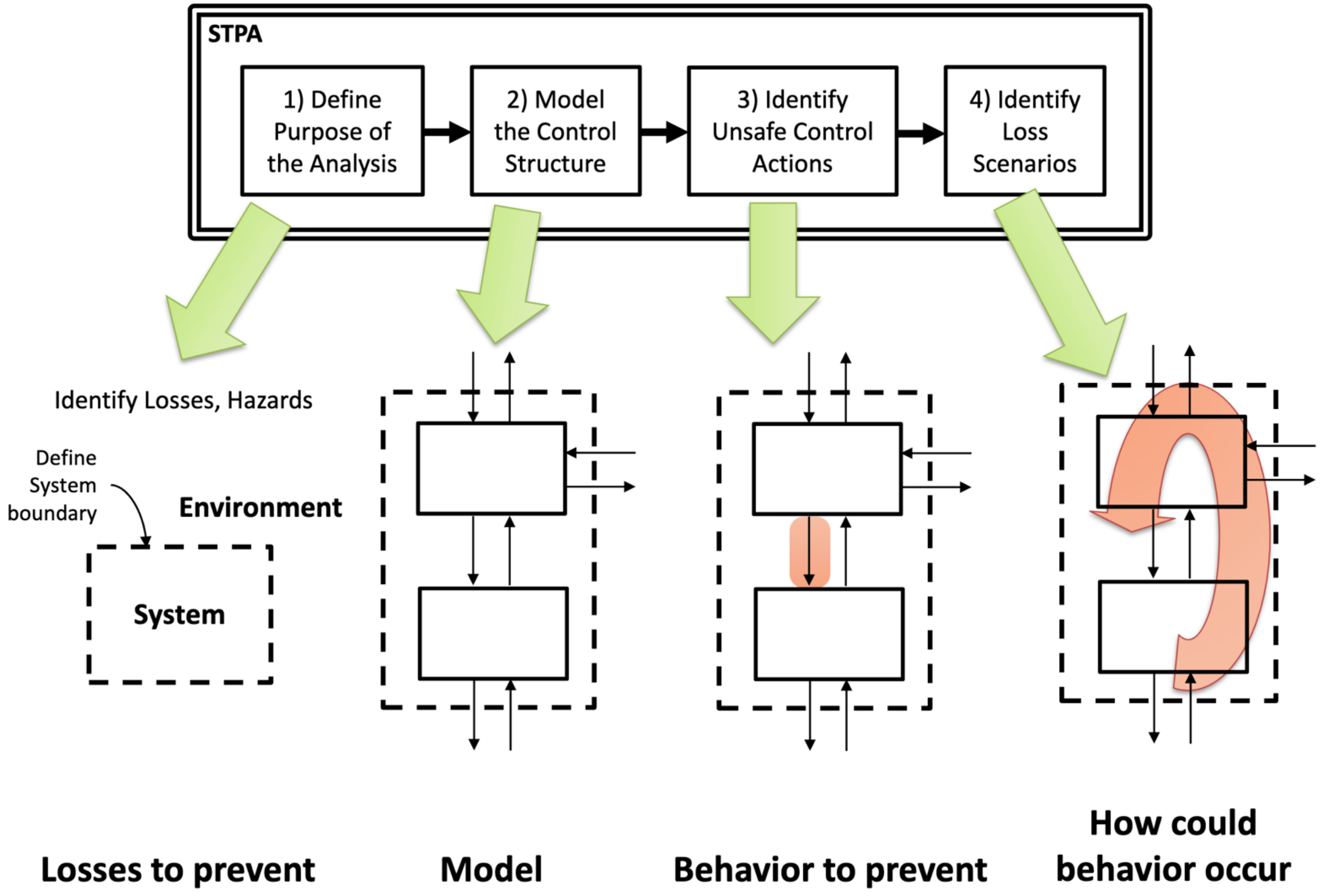
System Theoretic Process Analysis (STPA)

STPA is a technique for development and safety assessment

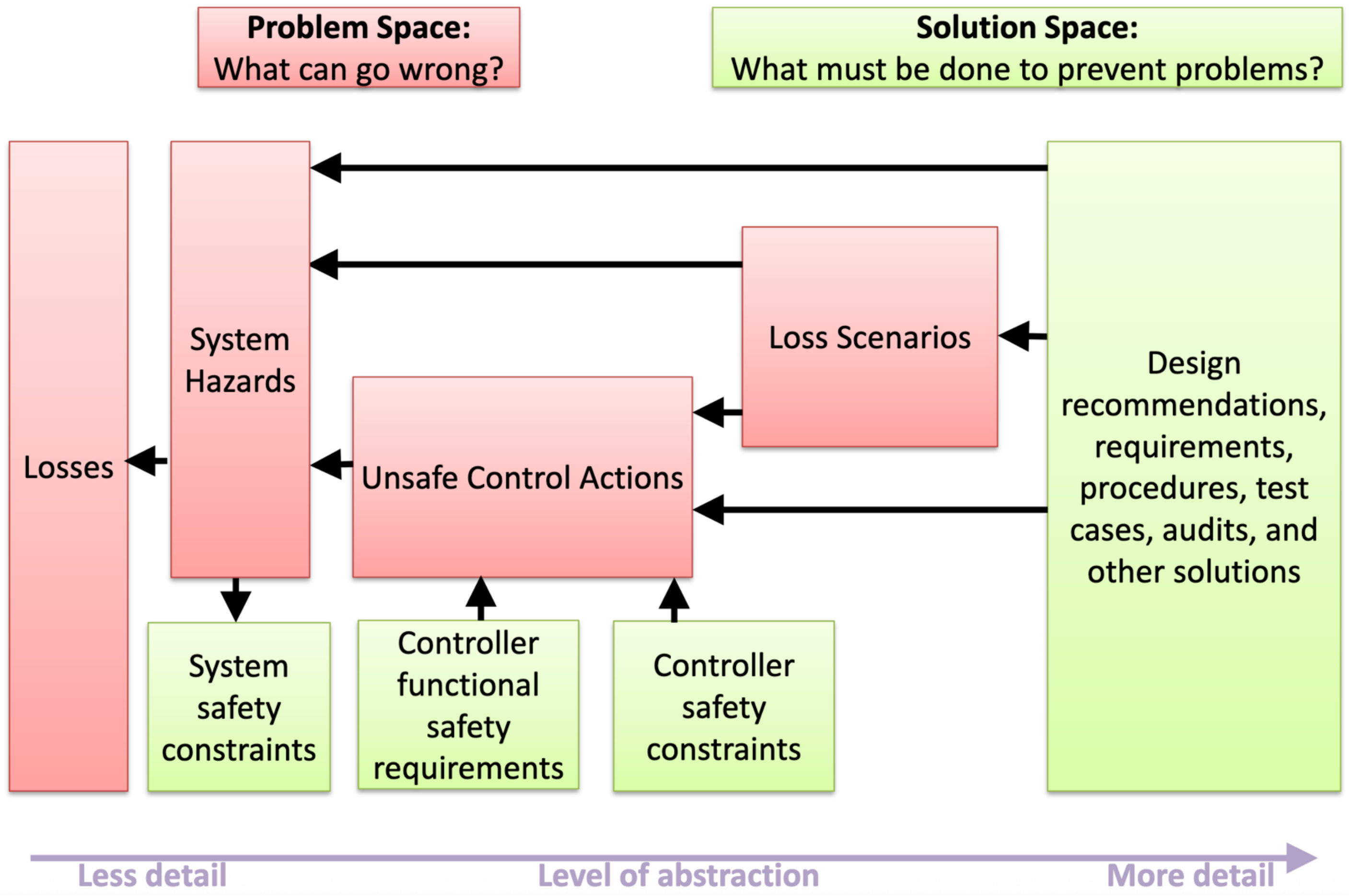
STPA can help anticipate hazardous scenarios caused by:

- Software, computers, and automation
- Human error/confusion
- System design errors
- Flawed assumptions
- Missing design requirements
- Interactions between systems

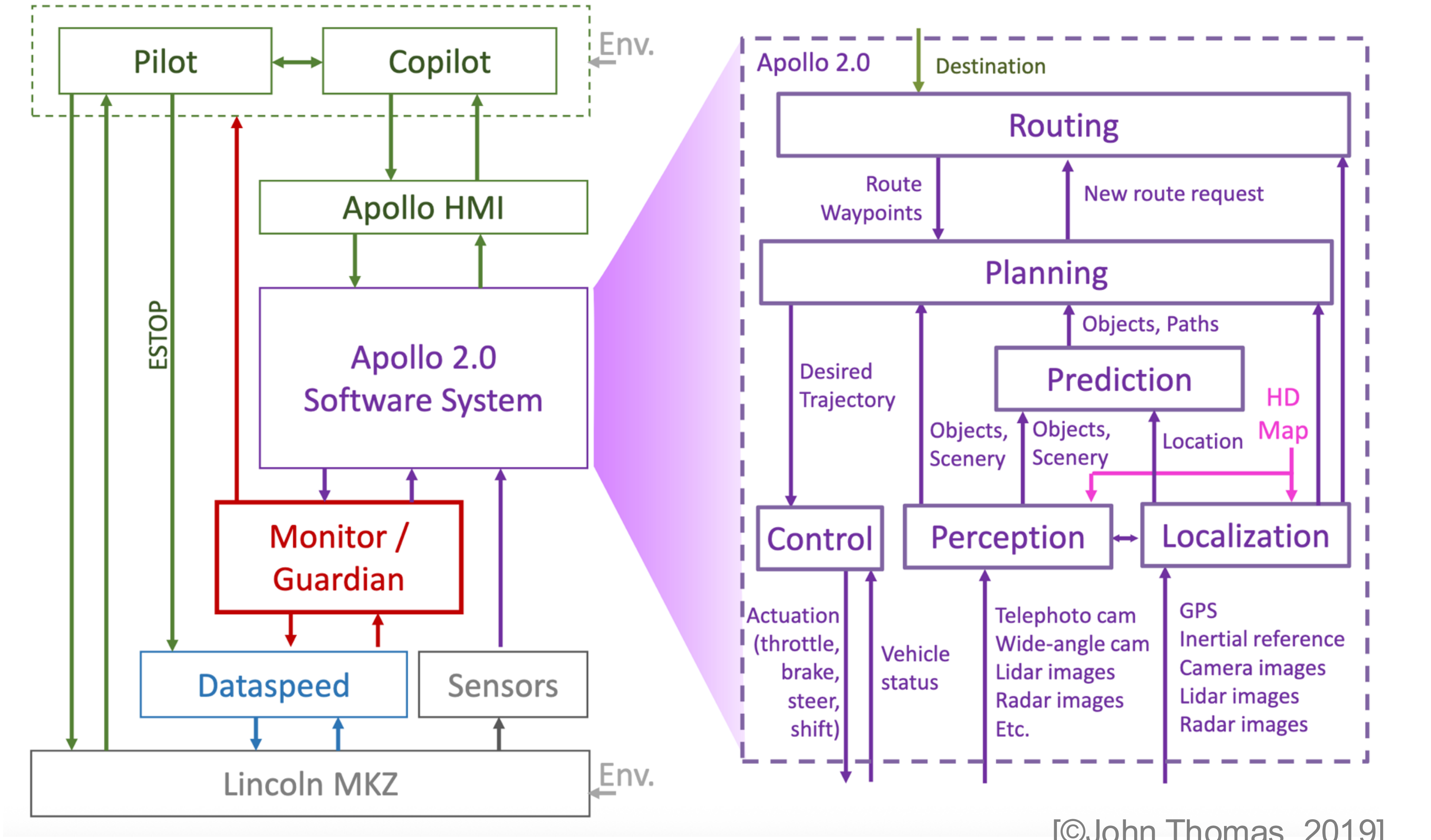
STPA Methodology (Leveson and Thomas)



Scenario-based Systems Thinking

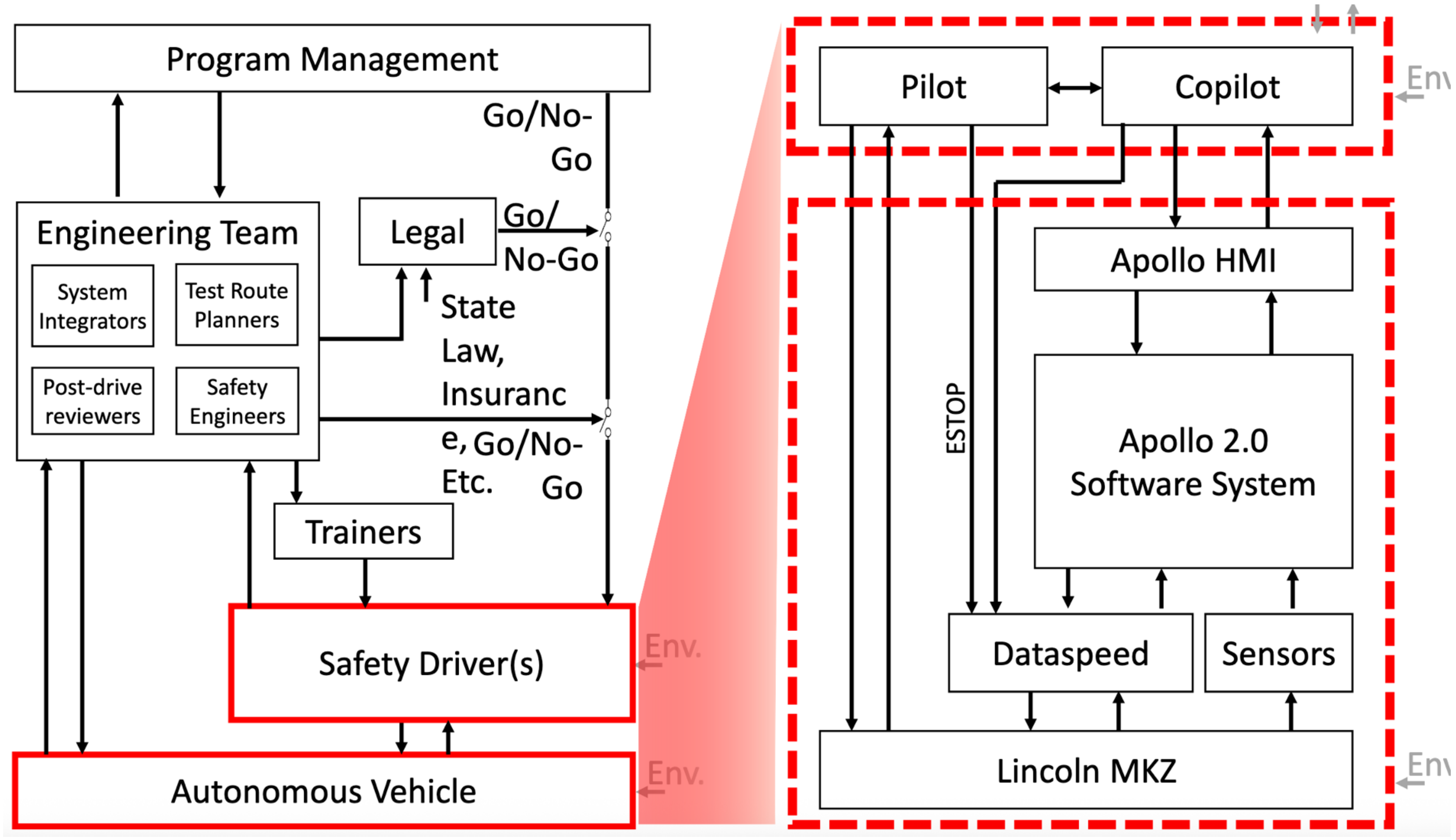

Problem Space:
What can go wrong?
Solution Space:
What must be done to prevent problems?
Losses
System Hazards
Loss Scenarios
Unsafe Control Actions
Design recommendations, requirements, procedures, test cases, audits, and other solutions
System safety constraints
Controller functional safety requirements
Controller safety constraints
Less detail
Level of abstraction
More detail

Looking at the full system: AV example



[©John Thomas, 2019]

Looking at the full system: AV example



[©John Thomas, 2019]

Safety Cases and Approaches to Assurance

Safety Case: How to Approach Assurance?

- As we saw, early attempts set in place prescriptive rules and safety codes to which adherence was mandatory
  - This includes standards, e.g., by ISO or SAE
- However, many engineered systems are so complex that this could rule out the entire intended operation if done in a heavy handed way
- Alternative: ask developers and operators to construct well reasoned arguments that their systems achieve acceptable levels of safety
- These arguments, together with supporting evidence, are typically referred to as a "safety case"

Safety Cases

- The purpose of a safety case:

  A safety case should communicate a clear, comprehensive and defensible argument that a system is acceptably safe to operate in a particular context

- Safety cases are already adopted in many industries, including defence, aerospace, railways and automotive sectors.
- Based on such practice, we can extract a few key attributes of what makes a good and useful safety case

Aspects of a Safety Case

- ‘**argument**’: the case exists to communicate an argument, to demonstrate how someone can reasonably conclude that a system is acceptably safe from the available evidence
- ‘**clear**’: it is a device for communicating ideas and information to a third party, e.g., regulator
- ‘**system**’: this could be anything from a network of pipes to a software module with parameters or operating procedures
- ‘**acceptably**’: In most applications, “absolute” safety is impossible. So, the case is argument to say how the system is safe enough (as per some notion of tolerable risk)
- ‘**context**’: most systems are only safe within a context of use, which should be defined in the safety case

Safety Case as a Physical Artifact

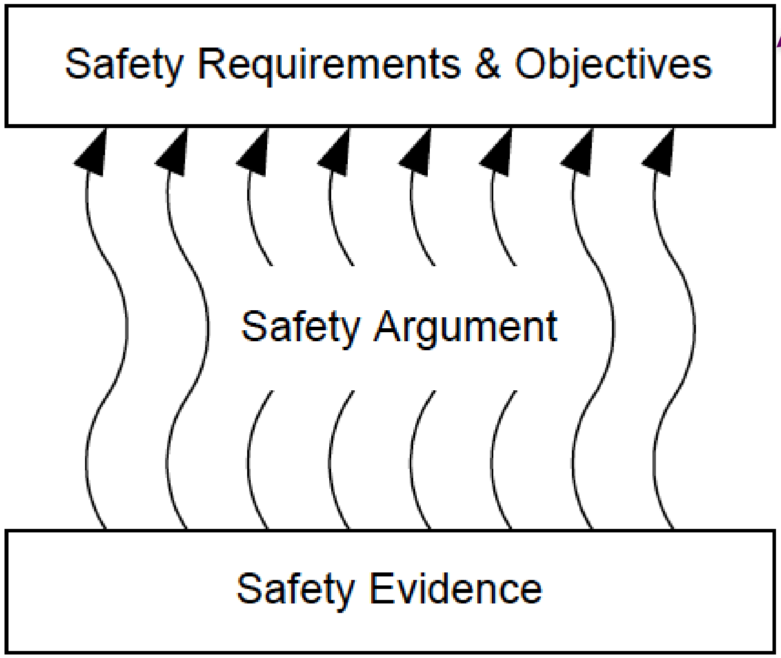
- Comprehensive and structured set of documentation
- To ensure safety can be demonstrated with reference to:
  - Arrangements and organisation, including emergency arrangements
  - Safety analyses
  - Compliance with standards and best practice
  - Acceptance tests
  - Audits and inspections
  - Feedback

Definition according to UK MoD

How to Argue?

[T. Kelly, SAE 04AE-149, 2003]

Domain analysis



Test Cases

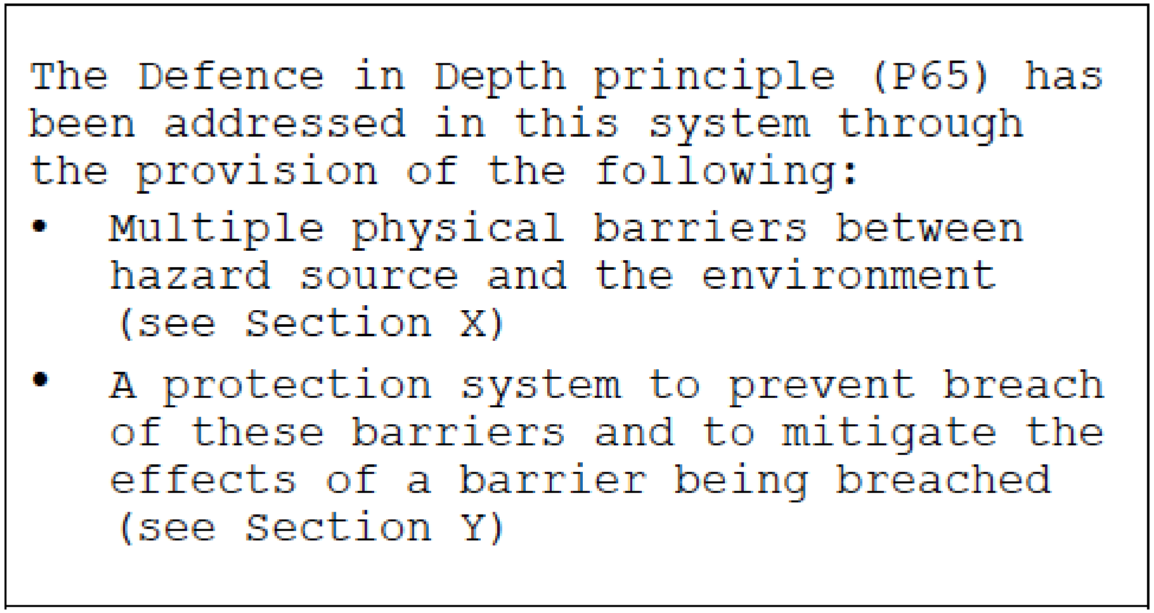
Model checking
& theorem proving

Physical component tests,
Unit tests, etc.

Acceptance test in design domain

Functional tests at system level

Communicating Safety Arguments: Typical Example in Textual Form



The Defence in Depth principle (P65) has been addressed in this system through the provision of the following:

- Multiple physical barriers between hazard source and the environment (see Section X)
- A protection system to prevent breach of these barriers and to mitigate the effects of a barrier being breached (see Section Y)

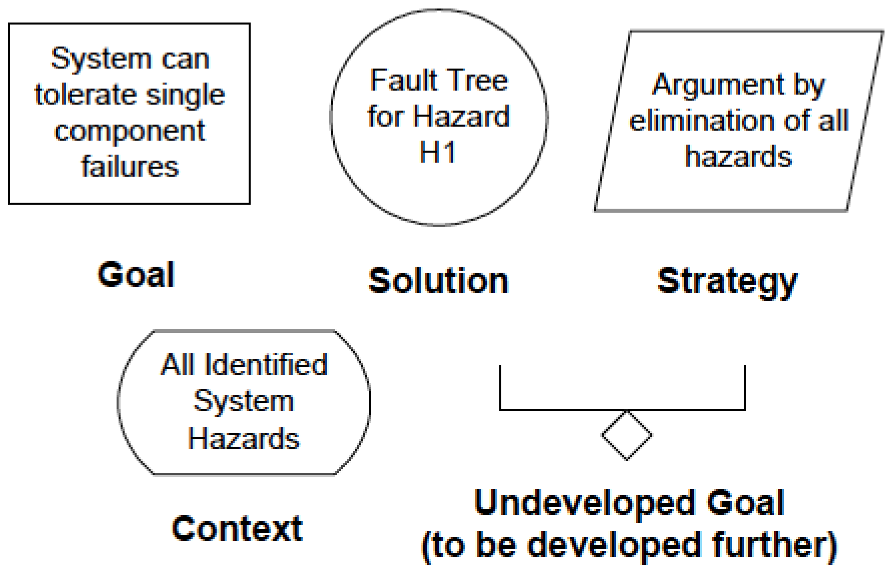
[T. Kelly, SAE 04AE-149, 2003]

… however, writing down long arguments can be both cumbersome and error-prone when teams of engineers work on such arguments

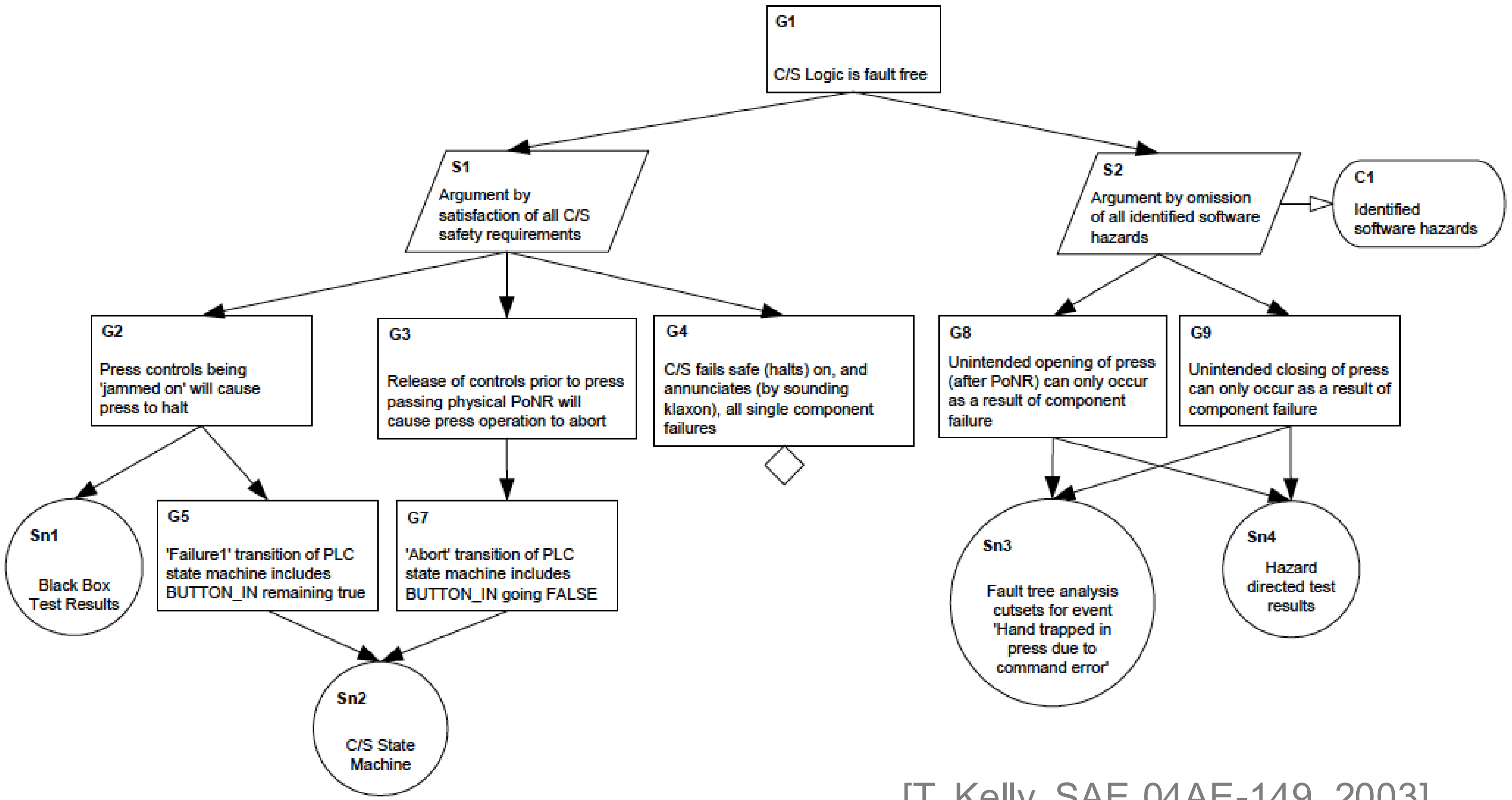
Formal notation: Goal-structuring Notation

GSN: a graphical argumentation notation - explicitly represents the individual elements of any safety argument

Vocabulary:

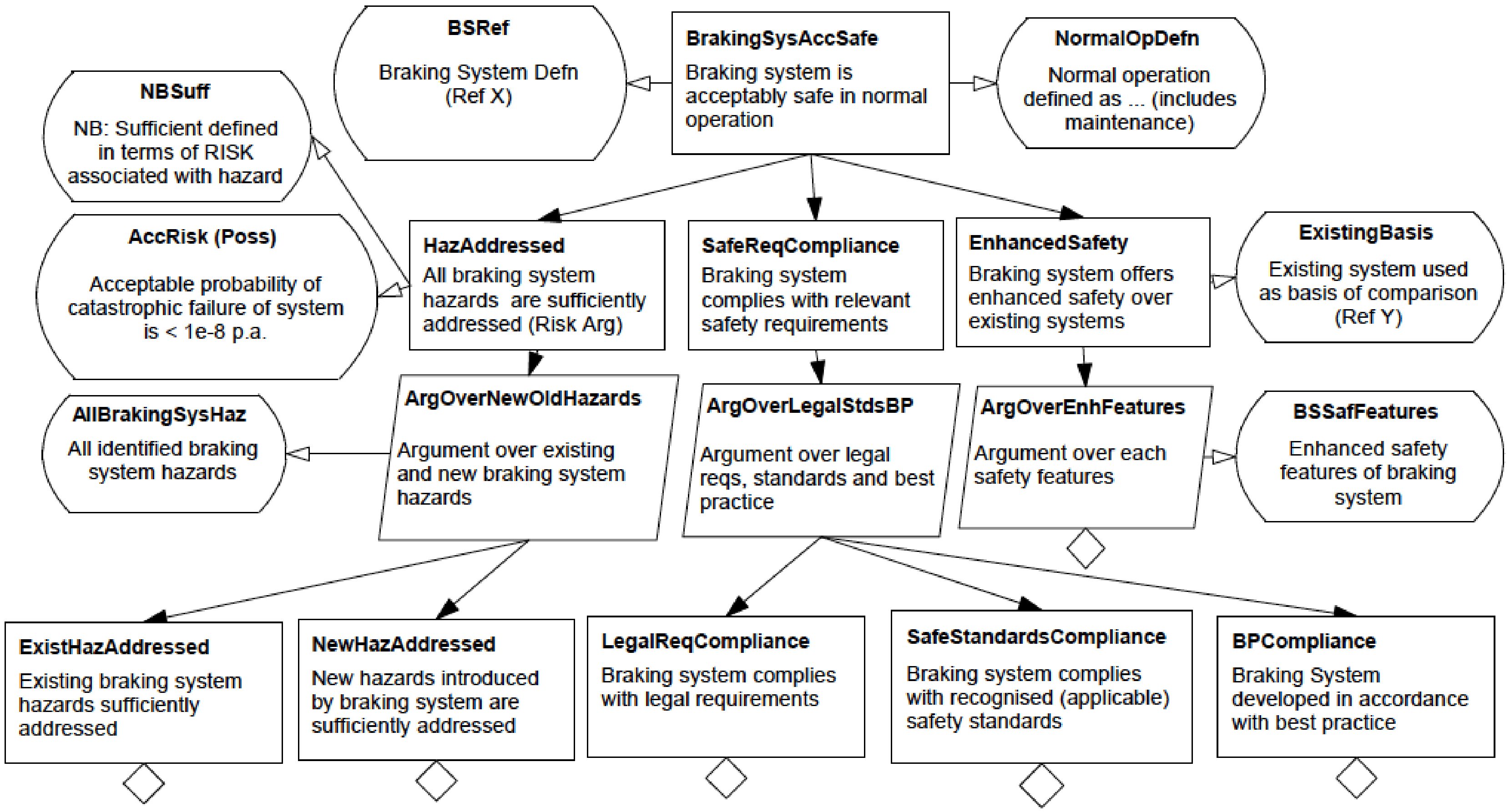

System can tolerate single component failures
Goal
Fault Tree for Hazard H1
Solution
Argument by elimination of all hazards
Strategy
All Identified System Hazards
Context
Undeveloped Goal (to be developed further)

Example Goal Structure for a Braking System


G1
C/S Logic is fault free
S1
Argument by satisfaction of all C/S safety requirements
S2
Argument by omission of all identified software hazards
C1
Identified software hazards
G2
Press controls being 'jammed on' will cause press to halt
G3
Release of controls prior to press passing physical PoNR will cause press operation to abort
G4
C/S fails safe (halts) on, and annunciates (by sounding klaxon), all single component failures
G8
Unintended opening of press (after PoNR) can only occur as a result of component failure
G9
Unintended closing of press can only occur as a result of component failure
Sn1
Black Box Test Results
G5
'Failure1' transition of PLC state machine includes BUTTON_IN remaining true
G7
'Abort' transition of PLC state machine includes BUTTON_IN going FALSE
Sn3
Fault tree analysis cutsets for event 'Hand trapped in press due to command error'
Sn4
Hazard directed test results
Sn2
C/S State Machine

[T. Kelly, SAE 04AE-149, 2003]

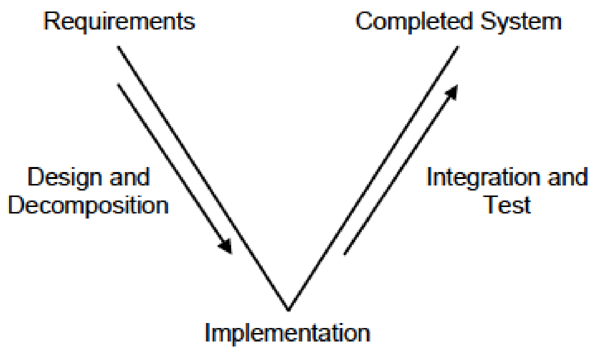
Sketch of a Preliminary Safety Argument


BSRef
Braking System Defn (Ref X)
BrakingSysAccSafe
Braking system is acceptably safe in normal operation
NormalOpDefn
Normal operation defined as ... (includes maintenance)
NBSuff
NB: Sufficient defined in terms of RISK associated with hazard
AccRisk (Poss)
Acceptable probability of catastrophic failure of system is < 1e-8 p.a.
HazAddressed
All braking system hazards are sufficiently addressed (Risk Arg)
SafeReqCompliance
Braking system complies with relevant safety requirements
EnhancedSafety
Braking system offers enhanced safety over existing systems
ExistingBasis
Existing system used as basis of comparison (Ref Y)
AllBrakingSysHaz
All identified braking system hazards
ArgOverNewOldHazards
Argument over existing and new braking system hazards
ArgOverLegalStdsBP
Argument over legal reqs, standards and best practice
ArgOverEnhFeatures
Argument over each safety features
BSSafFeatures
Enhanced safety features of braking system
ExistHazAddressed
Existing braking system hazards sufficiently addressed
NewHazAddressed
New hazards introduced by braking system are sufficiently addressed
LegalReqCompliance
Braking system complies with legal requirements
SafeStandardsCompliance
Braking system complies with recognised (applicable) safety standards
BPCompliance
Braking System developed in accordance with best practice

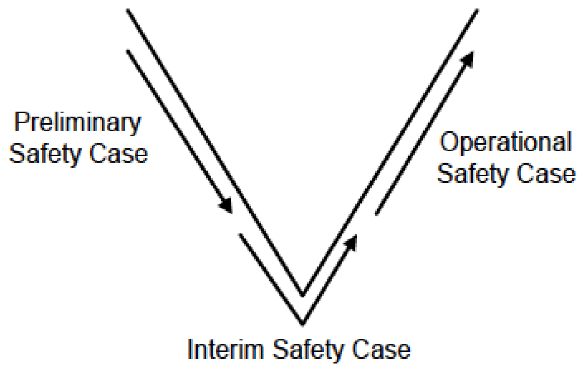
[T. Kelly, SAE 04AE-149, 2003]

Different Views of Development and Safety Lifecycles

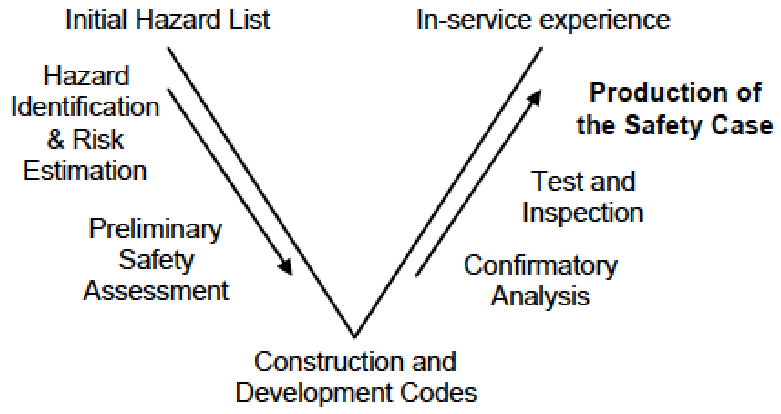
Design cycle:



Desired integrated Safety cycle:



Historical Safety cycle:

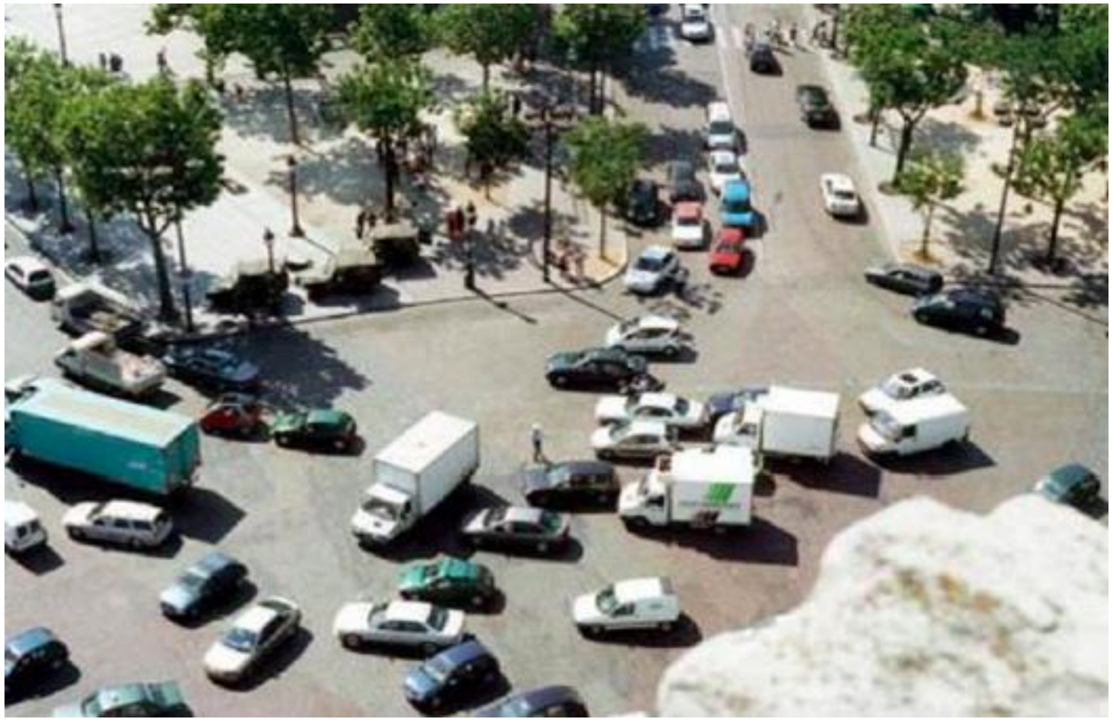

Initial Hazard List
In-service experience
Hazard Identification & Risk Estimation
Production of the Safety Case
Test and Inspection
Preliminary Safety Assessment
Confirmatory Analysis
Construction and Development Codes

[T. Kelly, SAE 04AE-149, 2003]

Responsibility, Blame and Implications for Safety Monitoring

Notions of Responsibility and Blame: Back to ADAS

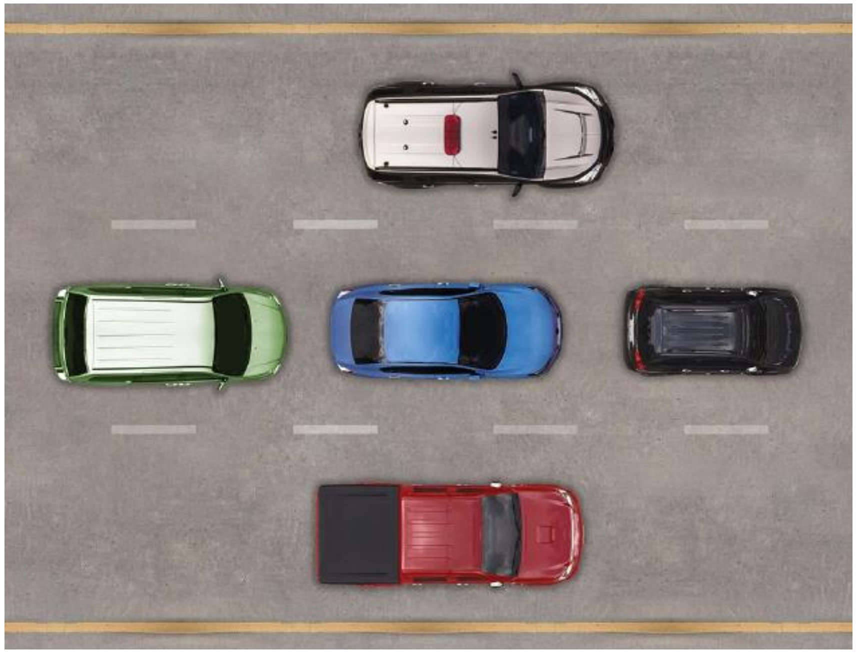
Why Difficult? Typical Operating Scenarios



Issues:

- Dynamic & Open Environments
- Incompleteness & Uncertainty (Model & Perception)
- Human in the loop (Social & Interaction Constraints)

Can We Ensure Safety, Always?



Example Question: Can the central car avoid all collisions? *

https://www.mobileye.com/technology/responsibility-sensitive-safety/

* [S. Shalev-Shwartz, S. Shammah, A. Shashua, On a formal model of safe and scalable self-driving cars. arXiv preprint arXiv:1708.06374, 2017]

What do we mean by *Safety*?

- Absolute Safety: An action a taken by a car is absolutely safe if no accident can follow it at some future time. This is impossible to achieve such as in the previous scenario
  - Forbidding the vehicle from ever being in such states could cause it not to drive at all
- What would be alternate ways of capturing the elements of human judgement? Traffic laws are well defined, but that may not be enough
- Duty of Care: Concept from Tort law, states that an individual should exercise “reasonable care” while performing acts that could harm others
  - Open to interpretation, so needs guidance for modelling

Responsibility-Sensitive-Safety (RSS)

- An attempt to formalize “duty of care”, by Shahua et al.
- Desiderata:
  - Sound: complies with human notions of law
  - Useful: not overly conservative
  - Efficiently verifiable: in the computational sense
- Formalize common sense rules such as:
  1. Do not hit someone from behind.
  2. Do not cut-in recklessly.
  3. Right-of-way is given, not taken.
  4. Be careful of areas with limited visibility
  5. If you can avoid an accident without causing another one, you must do it

Shashua’s Approach to Safety

In practice, the AV needs to know two things:

•Safe State: This is a state where there is no risk that the AV will cause an accident, even if other vehicles take unpredictable or reckless actions.

•Default Emergency Policy: This is a concept that defines the most aggressive evasive action that an AV can take to maintain or return to a Safe State.

They coin the term Cautious Command to represent the complete set of commands that maintains a Safe State. Set a hard rule that the AV will never make a command outside of the set of Cautious Commands. This ensures that the planning module itself will never cause an accident.

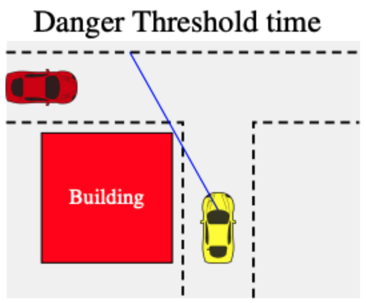
Example: Safe Longitudinal Distance

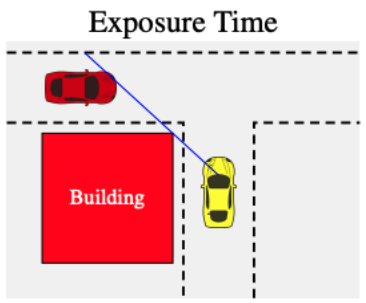
Safe Distance Formula

$$d_{\min} = L + T_f\left[v_r - v_f + \rho\left(a_a + a_b\right)\right] - \frac{\rho^2 a_b}{2} + \frac{(T_r - T_f)(v_r + \rho\, a_a - (T_f - \rho)a_b)}{2}$$

- L is the average length of the vehicles
- ρ is the response time of the rear vehicle
- v_r, v_f are the velocities of the rear/front vehicles
- a_a, a_b are the maximal acceleration/braking of the vehicles
- T_f is the time for the front car to reach a full stop if it would apply maximal braking
- T_r is the time for the rear car to reach a full stop if it would apply maximal acceleration during the response time, and from there on maximal braking

Specifying Constraints on Time/Velocity



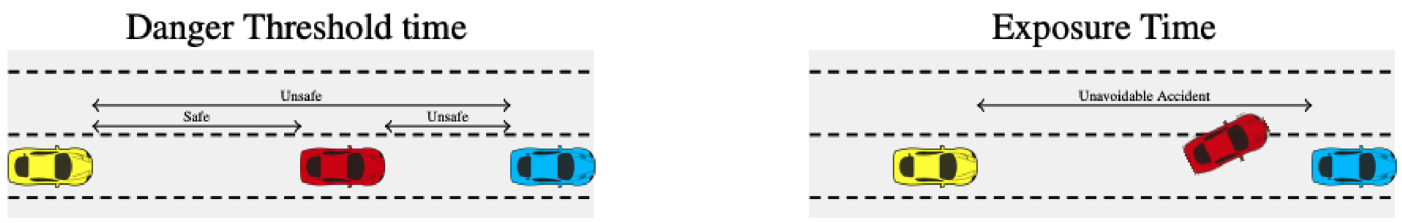


When do we cross "points of no return to safety"?

When does the obstacle become visible for the first time?

Consequences of Model: Responsibility

- We can give a pragmatic definition of responsibility
- An agent is *responsible* for an accident if it did not comply with the proper response constraints
- Analyse this:



- Yellow car is not responsible for the accident
- Red car did not respond properly to Blue car
- Red car was not involved in collision at all, can it be blamed?

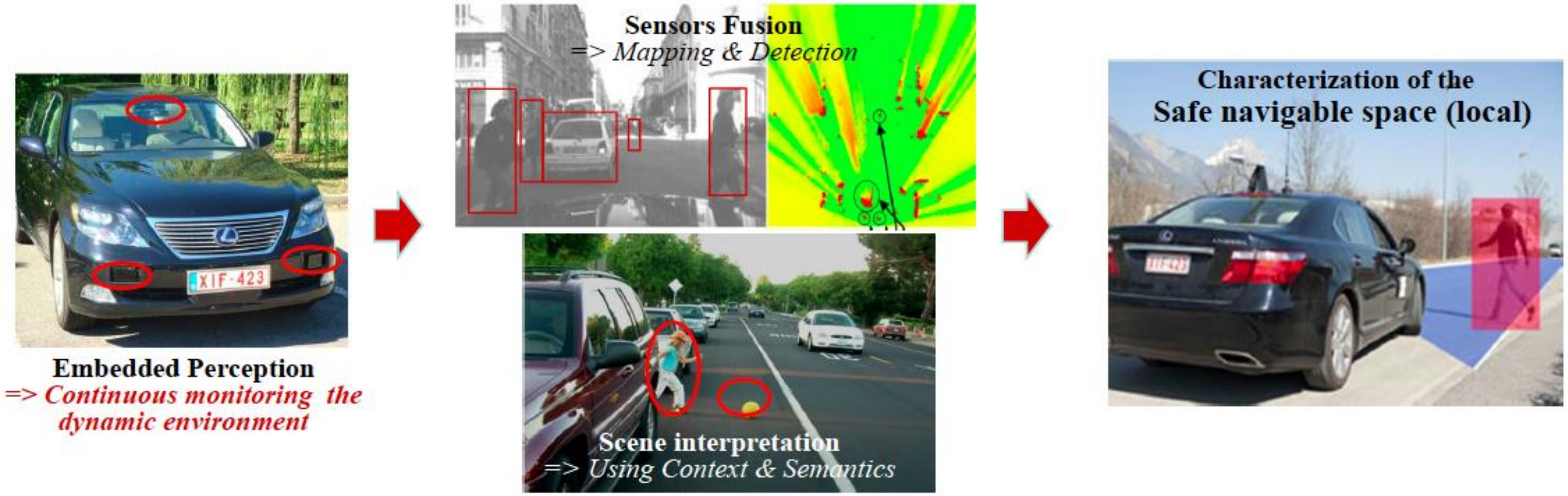
Consequences of Model: AI Utopia

- If all cars were controlled by RSS principles
- If all agents respect constraints of the form laid out in this framework
- It is possible for there to be driving scenarios where all basic response rules are universally enforceable

What are the limits of such reasoning?

1. Models of others
2. At an even lower level, perception system reliability

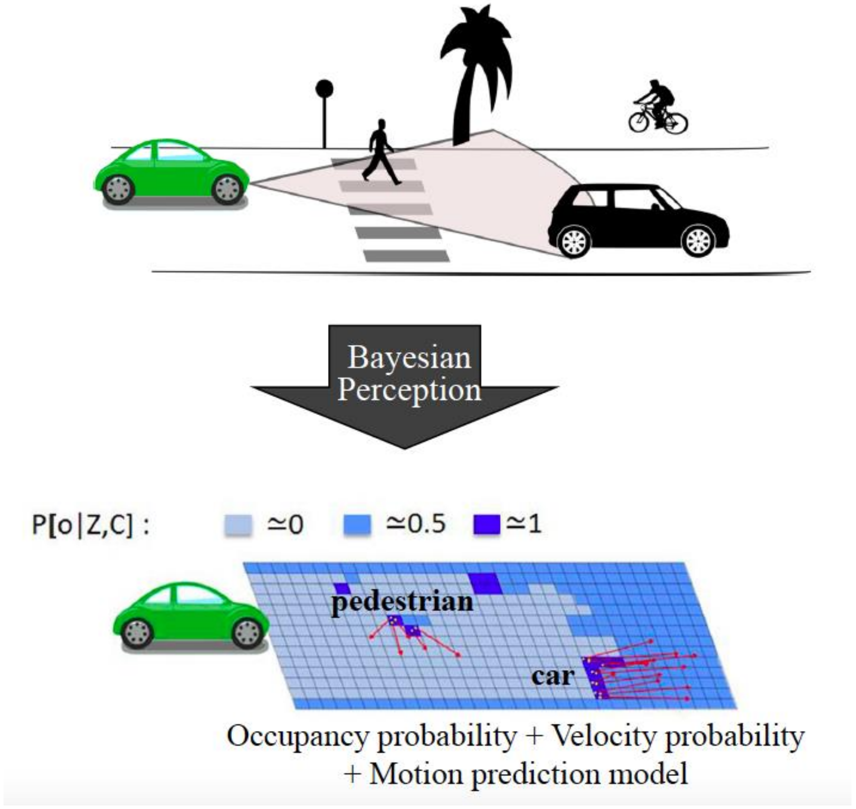
(Bayesian)
Perception is Inherently Uncertain


Embedded Perception
=> Continuous monitoring the dynamic environment
Sensors Fusion
=> Mapping & Detection
Scene interpretation
=> Using Context & Semantics
Characterization of the Safe navigable space (local)

[Source: C. Laugier et al.]

Typical Bayesian Perception Pipeline for ADAS/AV

- Estimate Spatial occupancy
- Analyze Motion Field (using Bayesian filtering)
- Initially, reason at the Grid level (no object segmentation, just prob. of occupancy, o, given observation z and state c)
- Then, register other objects on top of this data structure


Bayesian Perception
P[o|Z,C] :
≃0
≃0.5
≃1
pedestrian
car
Occupancy probability + Velocity probability + Motion prediction model

[Source: C. Laugier et al.]

Discussion Points

Some questions…

1. What are systems level issues arising in your own projects?

2. What would you need to be able to write safety cases for systems you develop?

3. Frameworks like RSS take a basic approach to responsibility and blame – what does that look like in your domains?